Experimental Validation of a Faithful Binary Circuit Model

Robert Najvirt, Matthias Függer, Thomas Nowak, Ulrich Schmid, Michael Hofbauer, Kurt Schweiger

Experimental Validation of a Faithful Binary Circuit Model

Robert Najvirt
ECS group, TU Wien
rnajvirt@ecs.tuwien.ac.at

Matthias Függer
MPI-INF
mfuegger@mpi-inf.mpg.de

Thomas Nowak
ENS Paris
thomas.nowak@ens.fr

Ulrich Schmid
ECS group, TU Wien
s@ecs.tuwien.ac.at

Michael Hofbauer
Inst. EMCE, TU Wien
michael.hofbauer@tuwien.ac.at

Kurt Schweiger
Avago Technologies
kurt.schweiger@avagotech.com

ABSTRACT

Fast digital timing simulations based on continuous-time, digital-value circuit models are an attractive and heavily used alternative to analog simulations. Models based on analytic delay formulas are particularly interesting here, as they also facilitate formal verification and delay bound synthesis of complex circuits.

Recently, Függer et al. (arXiv:1406.2544 [cs.OH]) proposed a circuit model based on so-called involution channels. It is the first binary circuit model that *realistically* captures solvability of short-pulse filtration, a non-trivial glitch propagation problem related to building one-shot inertial delays.

In this work, we address the question of whether involution channels also *accurately* model the delay of real circuits. Using both Spice simulations and physical measurements, we confirm that modeling an inverter chain by involution channels accurately describes reality. We also demonstrate that transitions in vanishing pulse trains are accurately predicted by the involution model. For our Spice simulations, we used both UMC-90 and UMC-65 technology, with varying supply voltages from nominal down to near sub-threshold range. The measurements were performed on a special-purpose UMC-90 ASIC that combines an inverter chain with low-intrusive high-speed on-chip analog amplifiers.

1. INTRODUCTION

Modern digital circuit design relies heavily on fast timing analysis techniques. In contrast to fully-fledged analog simulations, using Spice [1], for example, which are based on detailed analog models of all elements of a digital standard-cell library, several state-of-the-art tools use less detailed component models to speed-up simulation times. For example, the Synopsis CCS Timing model [2] characterizes the delay of a cell via input/output current waveforms given in tabular form, depending on parameters such as input slew rate and output capacitive load.

This research was partially funded by the Austrian Science Fund (FWF) projects SIC (P26436) and Rise (S11405).

GLSVLSI'15, May 20–22, 2015, Pittsburgh, PA, USA.
ACM 978-1-4503-3474-7/15/05.
http://dx.doi.org/10.1145/2742060.2742081.

Whereas such models facilitate a very *accurate* and reasonably fast timing simulation and functional correctness verification of a given circuit design, they are not suited for gaining a deeper understanding of the timing behavior of a complex circuit, its dependence on various parameters, and general performance and correctness characterizations of classes of circuits. To be more specific, consider some input-to-output delay characteristics Δ of a complex circuit, which typically depends on the input-to-output delay characteristics $\delta_1, \ldots, \delta_k$ of the constituent cells.

Determining Δ with empiric tabular-based delay models like the CCS Timing model inevitably introduces interpolation errors, discontinuities, cutoffs of unbounded function values, etc., which may not only lead to quantitatively wrong results, but may also predict qualitatively wrong circuit behaviors. Moreover, this approach admits only numerical solution methods for computing, e.g., fixed points in case of feedback-loops (see below).

A designer who tries to meet constraints on Δ thus would greatly benefit from having an accurate *analytic* expression $\Delta = f(\delta_1, \ldots, \delta_k)$ in terms of the constituent delays, which is sufficiently simple to highlight dependencies and may even facilitate a formal characterization of the solution space for Δ. This is particularly true at early design stages, where such knowledge may guide the search for alternative architectural designs.

As a long-term goal of our research, we hence target an *analytic framework* for the timing and correctness analysis of complex digital circuits. Such a framework, supported by tools, would not only allow for accurate performance and even power estimation [3, 4] at early design stages, but also pave the way to a thorough formal verification of a complex circuit's correctness. After all, a key concern for the latter is the detection of potential race conditions and hazardous glitches, which rests upon a rigorous timing analysis.

It is important to note, though, that statements about the correctness of a circuit *in a model* are meaningful only if they also imply correctness of the corresponding *real* circuit implementation. We hence call a model *realistic*, if a given problem can be solved in the model if and only if it can be solved by a real circuit, and *faithful* if it is both realistic and provides accurate timing predictions. Obviously, our ultimate target is a faithful analytic circuit model.

Unfortunately, determining analytic expressions for Δ, not to speak of characterizing its solution space, is complicated by the fact that the delay δ_i of a circuit's component i may depend on its input history and, hence, on the way it is used

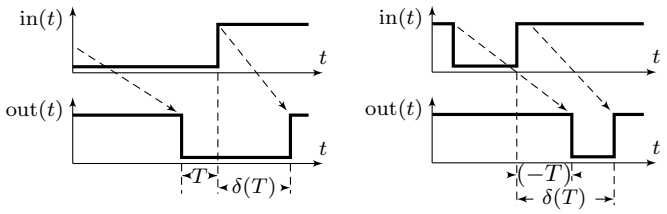

Figure 1: Left: Input/output signal of a single-history channel, involving the input-to-previous-output delay T and the resulting output-to-input delay $\delta(T)$. Right: Input transition with $T < 0$.

within the circuit. Even worse, in case of circuits involving feedback loops, computing Δ may involve solving a fixed-point problem, as δ_i may in fact depend on (the history of) Δ. Developing an analytic timing framework for complex circuits is hence a major challenge.

Particularly interesting for complex circuits are models that involve only discrete-valued, typically binary, continuous-time signals, as they facilitate a purely digital analysis framework. For example, the circuit model introduced in [5] combines zero-time Boolean gates with *single-history channels* that model circuit delays. Such channels are primarily characterized by a delay function δ that maps a transition occurring at the channel input at time t to its corresponding output transition at time $t + \delta(T)$, where T is the input-to-previous-output delay. Fig. 1 shows two examples. Note that single-history channels do not only allow to model decaying pulse propagation, but also vanishing pulses: If two succeeding input transitions would, according to $\delta(T)$, occur at the output in reversed order, they cancel each other. Further, single-history channels allow for different rising and falling transition delays, specified by two delay functions $\delta_\uparrow$ and $\delta_\downarrow$, respectively. Well-known instances of single-history channels are *pure* or *inertial delay* channels [6]; a more advanced example is the *Delay Degradation Model* (DDM) [7, 8] by Bellido-Díaz et al.

In [5], it was formally proved that no *bounded* single-history channel (i.e., where δ is also bounded from below) can be realistic, and hence faithful: For the simple *Short-Pulse Filtration* (SPF) problem, which is related to a circuit's ability to suppress a single glitch, the authors showed that every bounded single-history channel either contradicts the unsolvability of SPF in bounded time or the solvability of SPF in unbounded time in physical circuits.

In [9], however, Függer et al. provided what seems to be the first candidate for a realistic circuit model: It is based on *involution channels*, which are single-history channels with delay functions that are not bounded from below. Due to a continuity property of an involution channel output with respect to the presence of glitches at the input, which is due to the involution property, they proved that SPF can be solved precisely when this is possible with physical circuits.

Major contributions: The purpose of this paper is to experimentally explore the applicability of the involution model with respect to modern VLSI circuits. Our primary target is a chain of inverters, which allows to track the reshaping of pulse trains along the inverter stages. In order to extend the generality of our experimental results, we consider both two different VLSI technologies (UMC-65 nm and UMC-90 nm) as well as different supply voltages (from nominal down to close to the sub-threshold regime, which causes the delays to increase). Moreover, as we operate our inverter chains at their speed limits, we also employed dedicated experiments to validate the accuracy of (part of) our Spice simulations. For our validation measurements, we used a custom UMC-90 ASIC [10] containing an inverter chain monitored by low-intrusive high-speed on-chip analog amplifiers attached to a high-speed real-time oscilloscope. Comparing the measurement results with corresponding simulations (using the post-layout netlists extracted from the ASIC design) indeed showed a very good match.

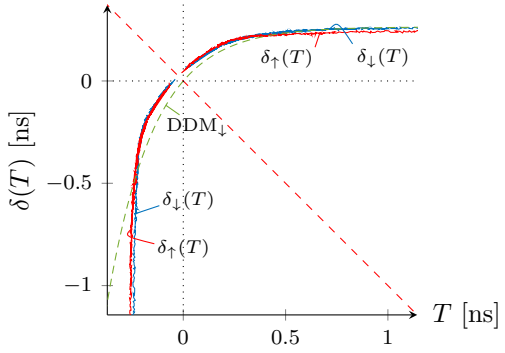

Figure 2: Measured $\delta_\downarrow$ (blue) and $\delta_\uparrow$ (red) for UMC-90 inverter chain for $V_{DD} = 0.6\ V$, which support the involution hypothesis. By contrast, there is no perfect fit for the exponential DDM delay function (dashed green).

Our experiments have been used to validate the following two features of the involution model:

(*Involution property*) By using input pulses of decreasing width, we empirically determine $\delta_\uparrow$ and $\delta_\downarrow$ for a single inverter. The resulting graphs for a supply voltage of $V_{DD} = 0.6\ V$ are depicted in Fig. 2. The involution property $-\delta_\downarrow(-\delta_\uparrow(T)) = T$ [9] has been used to extrapolate the functions' values for small T that do not allow direct delay measurements. The graphs support our claim that real delay functions are well approximated by involutions. By contrast, the exponential delay function of the DDM [8] is not an involution and cannot be fit to the experimental data over the whole range of T.

(*Good accuracy*) Using representative examples of pulse trains, we show that the involution model with the empirically determined $\delta_\uparrow$ and $\delta_\downarrow$ provides very good accuracy. We provide an explicit simulation algorithm for this purpose, which has also been implemented in ModelSim and thus allows timing simulations in our model. Fig. 3 shows an example pulse train (analog) together with the digital predictions from both the involution model and the DDM. We can see that the DDM both overly decays some short pulses (cp. the pulse at $t = 4$ ns in the middle waveform) and produces spurious pulses (bottom waveform at $t = 12$ ns).

2. RELATED WORK

Whereas there is a wealth of research devoted to the analog modeling of digital circuits (see [1, 11, 12, 13, 14] for a few references), none addressed the issue of characterizing delay functions with respect to solvability of problems. To the best of our knowledge, Függer et al. [9] have been the first to do so. As an underpinning of their involution model, they showed that, for any pair of involution delay functions

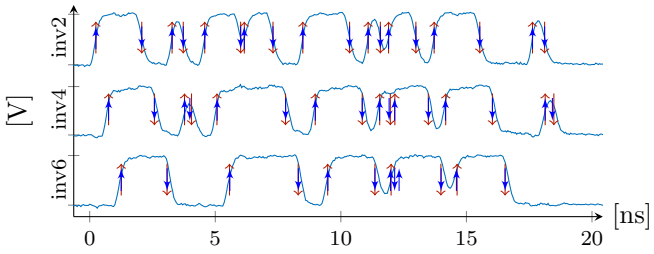

Figure 3: Measured waveform (solid) for the UMC-90 inverter chain, along with the corresponding predictions according to the involution model (red long up/down-arrows) (↑↓) and the DDM (blue short up/down arrows) (↑↓).

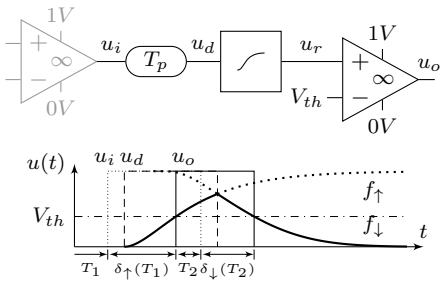

Figure 4: Simple analog channel model.

$\delta_\uparrow$, $\delta_\downarrow$, there are matching analog channel models consisting of a pure delay component with delay T_p, a slew-rate limiter with generalized switching waveforms $f_\uparrow$ and $f_\downarrow$, and a comparator with threshold V_{th}, as shown in Fig. 4.

On the other hand, digital circuit models have been proposed as a general approach for modeling asynchronous sequential switching circuits long time ago: Unger [6] introduced the well-known pure and inertial delay channels, which have been heavily used both in research and in industrial timing simulators. Bellido-Díaz et al. [8] proposed the PID model, and justified its appropriateness both analytically and by comparing model predictions against Spice simulations. In [15], the PID model (now called *Delay Degradation Model* DDM) was generalized from inverters to (N)AND and (N)OR gates. In the meantime, thanks to considerable efforts like [16, 15] spent on the question of how to extract the DDM model parameters from technology parameters, the DDM model has already made its way into digital timing analysis tools [7].

3. MODEL DESCRIPTION AND SIMULATION ALGORITHM

In this section, we briefly survey the cornerstones of the involution model and provide an explicit simulation algorithm, which iteratively constructs the execution, i.e., the traces of all signals, of a given circuit and trace of its inputs.

3.1 General model and simulation algorithm

The involution model rests on a binary-value continuous-time signal model. The state transitions of a signal are identified by *events* (t, x), where t is the event's time and x is the signal's new value. All state transition times are non-negative, but every signal has an initial value that is set at time $t = -\infty$. A signal can hence be represented by a (finite or infinite) sequence of events with increasing times and alternating values, whose first event is at time $-\infty$.

Algorithm 1 Initialization of the simulation algorithm

```
1:  for all input ports I do
2:      Pending(I) ← Events(I); Events(I) ← ∅
3:  end for
4:  for all gates B do
5:      (C_1, ..., C_d) ← Incoming(B)
6:      i ← B(Init(C_1), ..., Init(C_d))
7:      add (−∞, i) to Events(B)
8:  end for
9:  for all channels (B_1, B_2) do
10:     Last(B_1, B_2) ← (−∞, Init((B_1, B_2)))
11:     add Last(B_1, B_2) to Events((B_1, B_2))
12:     if Init(B_1) ≠ Init((B_1, B_2)) then
13:         add (0, Init(B_1)) to Pending((B_1, B_2))
14:     end if
15: end for
```

A circuit is a directed graph consisting of vertices that are either *input ports* or zero-time Boolean *gates*, and edges that are *channels*. Input ports have no incoming channels, every gate is assigned a Boolean function whose arity is equal to the number of its incoming edges. A gate can be declared to also form an *output port*, although they do not receive special treatment. Every channel is assigned a pair $(\delta_\uparrow, \delta_\downarrow)$ of delay functions; one for rising and one for falling transitions. Both gates and channels have a Boolean initial value.

The simulation algorithm takes as input a time T up to which the circuit should be simulated, and a sequence of events $\mathrm{Events}(I)$ up to time T for every input port I. Its output are sequences of events $\mathrm{Events}(C)$ for every component C (gate or channel) output of the circuit.

During the execution of the algorithm, it distinguishes *pending* and *fixed* events. Pending events are stored in the variable $\mathrm{Pending}(C)$, while fixed ones are transferred to $\mathrm{Events}(C)$. We write $\mathrm{Incoming}(B)$ for the ordered tuple of incoming edges of gate B and $\mathrm{Delay}(B_1, B_2)$ for the pair of delay functions of channel (B_1, B_2). Further, each channel (B_1, B_2) stores its last generated output event, whether it is canceled or not, in $\mathrm{Last}(B_1, B_2)$. We write $\mathrm{Init}(C)$ for the initial value of component C and we use the symbol f_B for the Boolean function of gate B.

Algorithm 1 performs the initializations needed for the following iterations of the simulation algorithm. Note that lines 12–14 produce a channel input event at time $t = 0$ if the initial values of a gate and an outgoing channel mismatch.

The main simulation algorithm is given in Algorithm 2. It uses function $\mathrm{Latest}(C, t)$ for a vertex (component) C and a time t, which is equal to the Boolean value of the most recent event for vertex C before or at time t. Note that both fixed and pending events are considered for $\mathrm{Latest}(C, t)$ and in line 14. When the loop terminates, the variables $\mathrm{Events}(C)$ contain the event sequences up to time T for all vertices C.

The algorithm proceeds by looking at the earliest pending events, declaring them as fixed, and propagating their effect to the other vertices via the channels. We highlight two noteworthy properties of the algorithm: (a) The delay $\delta(T)$ is a function of the input-to-previous output delay $T = t - t'$ (see line 15). (b) A pending output event of a channel is removed if a later input event causes an output event that occurs earlier (code line 17). In this case, the two events cancel at the channel output (pulse cancellation).

We implemented a custom VHDL involution channel module based on this simulation algorithm in Mentor Graphics

Algorithm 2 Timing prediction algorithm until time T

while there is a pending event at a time $\leq T$ **do**
  $t \leftarrow$ earliest time of a pending event
  for all comp. C with a pending event (t, x) at time t **do**
    move (t, x) from Pending(C) to Events(C)
  end for
  for all gates B **do**
    $(C_1, \ldots, C_d) \leftarrow$ Incoming(B)
    $v \leftarrow f_B\big(\text{Latest}(C_1, t), \ldots, \text{Latest}(C_d, t)\big)$
    add (t, v) to Events(B) **if** $v \neq$ Latest(B, t)
  end for
  for all channels (B_1, B_2) **do**
    $(\delta_\uparrow, \delta_\downarrow) \leftarrow$ Delay(B_1, B_2)
    if $\exists$ an event (t, x) in Events(B_1) at time t **then**
      $(t', x') \leftarrow$ Last(B_1, B_2)
      $\delta \leftarrow \delta_\uparrow(t - t')$ **if** $x = 1$ **and** $\delta \leftarrow \delta_\downarrow(t - t')$ **otherwise**
      Last(B_1, B_2) $\leftarrow (t + \delta, x)$
      if $t + \delta \leq t'$ **then**
        remove (t', x') from Pending((B_1, B_2))
      else
        add $(t + \delta, x)$ to Pending((B_1, B_2))
      end if
    end if
  end for
end while

ModelSim, which thereby supports digital timing simulations in the involution model.

3.2 Involution Channels

An involution channel is a channel whose pair of delay functions $(\delta_\downarrow, \delta_\uparrow)$ satisfies the following properties: Each delay function is defined on an open unbounded interval of the form $(z, +\infty)$ with $z \in \mathbb{R}$. The delay functions $\delta_\downarrow(T)$ and $\delta_\uparrow(T)$ are strictly increasing, concave, differentiable, and bounded as $T \to \infty$. Both $-\delta_\uparrow\big(-\delta_\downarrow(T)\big) = T$ and $-\delta_\downarrow\big(-\delta_\uparrow(T)\big) = T$ for all T in the respective domains. The domains are maximal in the sense that neither $\delta_\uparrow$ nor $\delta_\downarrow$ is bounded from below.

An involution channel is called *strictly causal* if $\delta_\downarrow(0) > 0$. By the above properties, this is equivalent to $\delta_\uparrow(0) > 0$.

Geometrically speaking, the properties for an involution channel say that $\delta_\uparrow$ is equal to $\delta_\downarrow$ reflected over the line $y = -x$. The defining property for strict causality means that the two curves meet in the second quadrant ($x \leq 0$, $y \geq 0$), at some point $(-\delta_{\min}, \delta_{\min})$ that gives the minimal delay $\delta_{\min}$ by which a non-canceled transition is propagated; Fig. 2 shows an example.

Függer et al. [9] proved that the simulation algorithm terminates with a unique, consistent collection of event sequences if all channels are strictly causal involution channels.

4. EXPERIMENTAL SETUP

The target of all our experiments is an inverter chain, which is a natural choice for validating involution channels. We consider two different bulk CMOS implementation technologies, namely, UMC-90 nm and UMC-65 nm. For UMC-65, we resorted to Spice simulations of a 7-stage inverter chain from a standard cell library. In case of UMC-90, we relied on a custom ASIC described in [10], which has been developed for on-chip measurements of single-event transient pulse shapes in VLSI circuits. It provides a 7-stage inverter chain built from 700 nm x 80 nm (W x L) pMOS and 360 nm x 80 nm nMOS transistors, with threshold voltages 0.29 V and 0.26 V, respectively, and a nominal supply voltage of

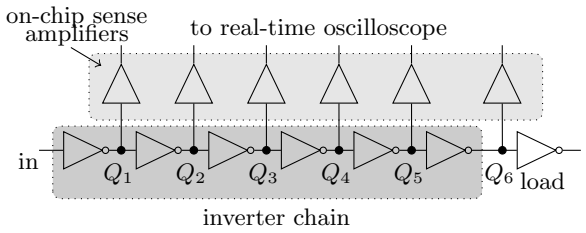

Figure 5: Schematics of the ASIC used for validation measurements. It combines an inverter chain with analog high-speed sense amplifiers.

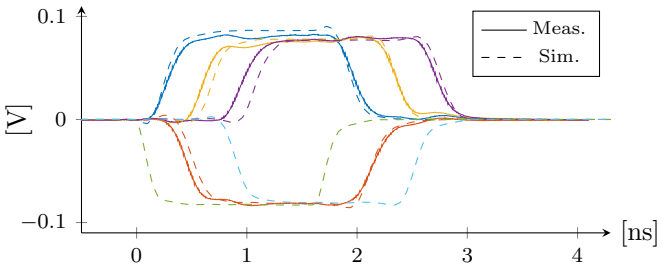

Figure 6: Measured (solid) vs. simulated (dashed) waveforms at Q_1, Q_3, Q_5 (bottom, shifted by $-V_{DD}$) and Q_2, Q_4, Q_6 (top), for $V_{DD} = 0.6\ V$. Note the amplifier gain of 0.15.

$V_{DD} = 1$ V. Detailed post-layout simulation models for the entire circuit are also available.

What makes this ASIC an ideal target for our experiments is the fact that all inverter outputs are connected to on-chip low-intrusive high-speed analog sense amplifiers, as shown in Fig. 5. As the amplifiers can drive the 50 Ω input of a high-speed real-time oscilloscope, they allow faithful analog recording of the output signal waveforms. The amplifiers have a measured gain of 0.15 with an overall -3 dB cutoff frequency of approximately 8.5 GHz, and constitute an additional (low) load approximately equivalent to 3 inverter inputs. Independent power supplies and grounds for inverters and amplifiers facilitate measurements with different digital supply voltages V_{DD}. According to [10], extensive simulations and measurements have revealed excellent measurement accuracy (within the linear bandwidth of the amplifiers).

The availability of both the ASIC and the corresponding post-layout simulation model allowed us to validate our simulation results: Given that we operate the inverters at their bandwidth limits, it is important to make sure that they indeed match reality.

For our measurements, we bonded the ASIC directly to an RF-substrate PCB (Rogers RO 4350), which routes the pins to dedicated SMP connectors. The PCB with the ASIC itself is mounted on a Peltier-cooled copper heat sink, which guarantees a stable operating temperature. In our final measurement setup, four high-performance RF cables of identical length connect the SMP outputs of the amplifiers attached to inverters 2,3,4, and 6 to the inputs of a 4-channel 12 GHz Tektronix TDS 6154B real-time oscilloscope, which records the waveforms with a sampling rate of 20 GS/s. The input of the first inverter is directly driven by a 3.35 GHz Agilent 81134A pulse/pattern generator. The entire setup was optimized to minimize distortions of the pulse shapes.

Validation measurements have been conducted for digital supply voltages in the range $V_{DD} \in \{0.3 \ldots 1\}$ V, which result in inverter output waveforms within the linear operation range of the sense amplifiers. In order to "hide" the much better driving capabilities of the pulse generator, we only considered inverters 3–6 in our measurements, that is, the output of inverter 2 was considered as the actual input of the chain. Note that the output bias due to AC coupling and the gain of the sense amplifiers had to be compensated when evaluating our results; the (identical) amplifier and cable delays automatically cancel out for stage to stage delay measurements.

The comparison of the measured waveforms with the ones obtained in corresponding analog simulations showed a very good match, see Fig. 6 for an example. As a consequence, we can reasonably infer that Spice simulations faithfully represent reality also in situations where no measurement results are available: For UMC-90 in settings near $V_{DD} = 1\ V$, where the measurement results are inaccurate due to amplifier bandwidth limitations, and for UMC-65 in general, as there was no measurement ASIC available.

All post-processing, including averaging, curve fitting and extraction of the delay functions based on threshold crossing times, was done in Matlab.

5. INVOLUTION VALIDATION

For determining $\delta_\downarrow(T)$ of a single inverter output channel, we used a sweep of 1-0-1 pulses with different widths at the inverter input. Analogously, $\delta_\uparrow(T)$ was determined using 1-0-1 input pulses. In order to mitigate noise and measurement errors, we computed the average of 1000 single pulse experiments for every measured pulse width.

For values $T \geq \delta_{\min}$, where no output cancellation takes place, both T and $\delta(T)$ can be directly observed from the input/output waveforms by determining the points in time where the threshold voltage V_{th} was crossed. The delay $\delta_\downarrow(T)$, e.g., was computed as the difference between the threshold crossing of the falling output and that of the rising input. T as the time from the latter to the threshold crossing of the rising output, i.e., the previous output transition (cp. Fig. 4). Note that we assume the same logical threshold voltage V_{th} for both rising and falling transitions, as in Fig. 4.

Measuring $\delta_\downarrow(T)$ and $\delta_\uparrow(T)$ for $T < \delta_{\min}$ is impossible, however, as such transitions are not propagated to the output: Expressed in terms of the analog waveforms, this corresponds to the situation where the output never reaches the threshold voltage V_{th}. We hence used the involution property $\delta_\downarrow(T) = -\delta_\uparrow(-T)$ established in [9] to extrapolate these values for $\delta_\downarrow$ from $\delta_\uparrow$ and vice versa.

Fig. 2 already provided in Sec. 1 shows the delays $\delta_\downarrow$ and $\delta_\uparrow$ drawn from the measurements of inverter 3 (which has Q_2 as its input and Q_3 as its output) in the chain depicted in Fig. 5, with $V_{DD} = 0.6$ V. For this supply voltage, we are sure that the sense amplifiers are within their linear operating range. Similar results have been obtained for all reasonable values of V_{DD}, both for UMC-90 and UMC-65: Fig. 7 shows $\delta_\downarrow$ drawn from the measurements and one drawn from simulations of inverter 3 in the UMC-90 inverter chain, Fig. 8 shows some simulated $\delta_\downarrow$ for UMC-65.

Our results clearly support our claim that real delay functions are well-approximated by means of involutions. By contrast, it is impossible to fit the exponential delay of the DDM [8] over the whole range of T: Fig. 2 and Fig. 8 also

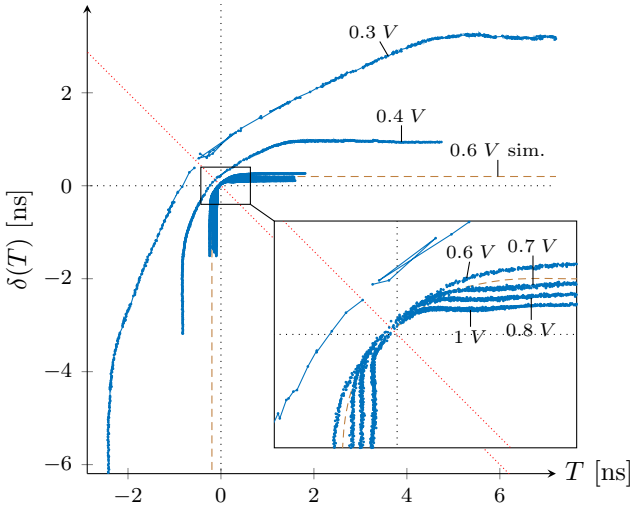

Figure 7: Measured $\delta_\downarrow$ for UMC-90 inverter chain for $V_{DD} \in \{0.3, 0.4, 0.6, 0.7, 0.8, 1\}$ V and simulated (dashed brown) $\delta_\downarrow$ for $V_{DD} = 0.6$ V.

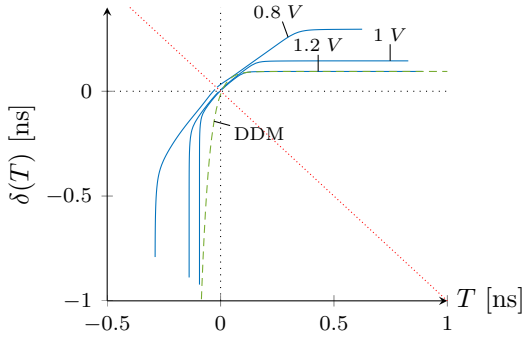

Figure 8: Simulated $\delta_\downarrow$ for UMC-65 inverter chain for $V_{DD} \in \{0.8, 1, 1.2\}$ V. DDM fitting for $V_{DD} = 1.2\ V$.

show the exponential delay function of the DDM for parameters T_0 and $\tau = (T_{95} - T_0)/3$ chosen according to Eq. (5) in [8], where T_{95} is such that $\delta(T_{95}) = 0.95\delta_\infty$. The parameter T_0 has been determined empirically to fit the measured delays best: For $V_{DD} = 0.6\,V$, we obtained $T_0 = 0$ ps and $T_{95} = 684$ ps (falling transitions) resp. $T_{95} = 527$ ps (rising) for UMC-90 and $T_0 = 5$ ps and $T_{95} = 115$ ps (falling) resp. $T_{95} = 151$ ps (rising) for UMC-65. It is apparent that there is a substantial mismatch for small values of the input-to-previous-output delay T for both UMC-90 and UMC-65.

6. MODELING ACCURACY

We next complement the results obtained in Sec. 5, which focused on the validation of the involution property, by experiments devoted to the accuracy of the involution model. More specifically, using the empirically determined functions $\delta_\uparrow$ and $\delta_\downarrow$ as well as the fitted exponential delay function of the DDM in the simulation algorithm Alg. 2, we compare the simulated/measured analog output waveforms obtained for certain input pulse trains with the corresponding model predictions. Our results reveal that the modeling accuracy is very good in general, and that the involution model outperforms the DDM in scenarios with short pulses.

Fig. 3 already provided in Sec. 1 shows the measured waveforms for the UMC-90 inverter chain (outputs Q_2, Q_4 and

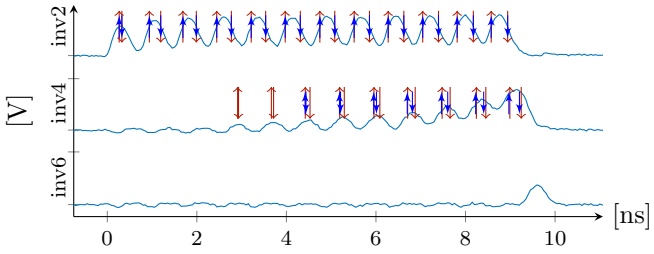

Figure 9: Measured fixed-frequency 1:1.31 duty cycle pulse train (solid) for the UMC-90 inverter chain, along with the corresponding predictions according to the involution model (red long up/down-arrows) (↑↓) and the DDM (blue short up/down arrows) (↑↓).

Q_6 in Fig. 5) when stimulated with a pulse train that incorporates pulses of various lengths. The crossings of the waveforms with the imaginary horizontal line at $V_{th} = 0.5V_{DD}$ give the times when the real circuit would change state. The long (red) up/down-arrows represent the signal transitions predicted by the involution model, the short (blue) up/down arrows give the DDM predictions. Whereas both models provide accurate predictions in most cases, the DDM tends to overly decay short pulses like the one at $t = 4$ ns in the middle waveform. In addition, the DDM produces spurious pulses (that do not exist in the real waveform), as can be seen e.g. in the bottom waveform at $t = 12$ ns. Similar results have been obtained for UMC-65.

We also investigated the model predictions in the case of a high frequency pulse train with a 0:1 duty cycle of 1:1.31, which captures the accumulation of energy even from non-propagated pulses. They are presented in Fig. 9, which again shows the measured waveforms for our UMC-90 inverter chain. Whereas both the DDM and the involution model accurately predict the output Q_2 (top waveform), both generate spurious pulses (before $t = 8$ ns) in the case of Q_4 (middle waveform), but not for Q_6 (bottom waveform). We conjecture that this inaccuracy is the price for the simplicity of any single-history model, which can only rely on the input-to-previous-output delay T in its predictions.

7. CONCLUSIONS

We experimentally evaluated the accuracy of the involution channel model, which is the first candidate for a realistic circuit model proposed so far. Using simulations of an inverter chain in both UMC-90 nm and UMC-65 nm technology, as well as measurements using a special-purpose ASIC, we could strengthen the hypothesis that real circuit delays are indeed well-approximated by involutions. Moreover, the model predictions obtained by using the empirically determined involution delay functions in the timing simulation algorithm match the analog waveforms quite accurately.

Whereas these findings further support the hypothesis that involution channels are indeed a promising candidate for the first faithful circuit model known so far, important issues are left for further research. Among these is the question of how to efficiently determine the actual involutions for a given circuit technology.

8. REFERENCES

[1] L. W. Nagel and D. Pederson, "SPICE (Simulation Program with Integrated Circuit Emphasis)," Tech. Rep. UCB/ERL M382, EECS Department, University of California, Berkeley, 1973.

[2] Synopsis, "CCS timing." Technical white paper v2.0, 2006.

[3] F. Najm, "A survey of power estimation techniques in vlsi circuits," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 2, no. 4, pp. 446–455, 1994.

[4] M. Favalli and L. Benini, "Analysis of glitch power dissipation in cmos ics," in *Proceedings of the 1995 international symposium on Low power desig n*, ISLPED '95, (New York, NY, USA), pp. 123–128, ACM, 1995.

[5] M. Függer, T. Nowak, and U. Schmid, "Unfaithful glitch propagation in existing binary circuit models," in *Proceedings 19th IEEE International Symposium on Asynchronous Cir cuits and Systems (ASYNC'13)*, pp. 191–199, IEEE Computer Society, 2013.

[6] S. H. Unger, "Asynchronous sequential switching circuits with unrestricted input changes," *IEEE ToC*, vol. 20, no. 12, pp. 1437–1444, 1971.

[7] M. J. Bellido-Díaz, J. Juan-Chico, and M. Valencia, *Logic-Timing Simulation and the Degradation Delay Model*. London: Imperial College Press, 2006.

[8] M. J. Bellido-Diaz, J. Juan-Chico, A. Acosta, M. Valencia, and J. L. Huertas, "Logical modelling of delay degradation effect in static cmos gates," *Circuits, Devices and Systems, IEE Proceedings* -, vol. 147, no. 2, pp. 107–117, 2000.

[9] M. Függer, R. Najvirt, T. Nowak, and U. Schmid, "Faithful glitch propagation in binary circuit models," *arXiv:1406.2544*, 2014. (appears in Proc. DATE'15).

[10] M. Hofbauer, K. Schweiger, H. Dietrich, H. Zimmermann, K.-O. Voss, B. Merk, U. Schmid, and A. Steininger, "Pulse shape measurements by on-chip sense amplifiers of single event transients propagating through a 90 nm bulk CMOS inverter chain," *IEEE Transactions on Nuclear Science*, vol. 59, pp. 2778–2784, Dec. 2012.

[11] M. A. Horowitz, *Timing Models for MOS Circuits*. PhD thesis, Stanford University, 1984.

[12] T.-M. Lin and C. Mead, "Signal delay in general RC networks," *IEEE TCAD*, vol. 3, no. 4, pp. 331–349, 1984.

[13] L. Pillage and R. Rohrer, "Asymptotic waveform evaluation for timing analysis," *IEEE TCAD*, vol. 9, no. 4, pp. 352–366, 1990.

[14] A.-C. Deng and Y.-C. Shiau, "Generic linear RC delay modeling for digital CMOS circuits," *IEEE TCAD*, vol. 9, no. 4, pp. 367–376, 1990.

[15] J. Juan-Chico, M. J. Bellido, P. Ruiz-de Clavijo, A. J. Acosta, and M. Valencia, "Degradation delay model extension to CMOS gates," in *Integrated Circuit Design*, LNCS 1918, pp. 149–158, Springer, 2000.

[16] A. Millan, J. Juan, M. J. Bellido, P. Ruiz-de Clavijo, and D. Guerrero, "Characterization of normal propagation delay for delay degradation model (DDM)," in *Integrated Circuit Design*, LNCS 2451, pp. 477–486, Springer, 2002.